

[illegible][illegible]

```

LL          IIIII
LL          I.IIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLLL
LLLLLLLLLLL

            IIIII
            I.IIII
            II
            II
            II
            II
            II
            II
            II
            II
            II
            II
            II
            II
            IIIII
            IIIII

SSSSSSSSS
SSSSSSSSS
SS
SS
SS
SS
SSSSSSS
SSSSSSS
SS
SS
SS
SS
SSSSSSSSS
SSSSSSSSS

```

```
0001 0 MODULE LIB_FILEIO ( ! Routines to read/write files
0002 0     LANGUAGE (BLISS32),
0003 0     IDENT = 'V04-000'
0004 0 ) =
0005 1 BEGIN
0006 1
0007 1
0008 1 *****
0009 1 *
0010 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 *  ALL RIGHTS RESERVED.
0013 1 *
0014 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 *  TRANSFERRED.
0020 1 *
0021 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 *  CORPORATION.
0024 1 *
0025 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *****
0029 1
0030 1
0031 1 ++
0032 1
0033 1 FACILITY: Library command processor
0034 1
0035 1 ABSTRACT:
0036 1
0037 1     The VAX/VMS librarian is invoked by DCL to process the LIBRARY
0038 1     command. It utilizes the librarian procedure set to perform
0039 1     the actual modifications to the library.
0040 1
0041 1 ENVIRONMENT:
0042 1
0043 1     VAX native, user mode.
0044 1
0045 1 --
0046 1
0047 1
0048 1 AUTHOR: Benn Schreiber,      CREATION DATE: 20-June-1979
0049 1
0050 1 MODIFIED BY:
0051 1
0052 1     V03-003 GJA0083      Greg Awdziejewicz      12-Apr-1984
0053 1     Make read errors in Get_record routine fatal, to
0054 1     stop infinite looping in the four Inputxxx modules
0055 1     when RMS can't read the file. (QAR #2076)
0056 1
0057 1     V03-002 MLJ0086      Martin L. Jack, 6-Apr-1982 13:19
```


:	58	0058	1	:		Fix V02-011 to work for shareable image libraries.
:	59	0059	1	:		
:	60	0060	1	:	V02-011	RPG0011 Bob Grosso 30-Mar-1982
:	61	0061	1	:		Use SQO (sequential only) option in FAB FOP to gain
:	62	0062	1	:		network performance enhancement by restricting file
:	63	0063	1	:		to sequential processing.
:	64	0064	1	:		
:	65	0065	1	:	V02-003	RPG0003 Bob Grosso 07-Jan-1982
:	66	0066	1	:		Return error when input file locked.
:	67	0067	1	:		
:	68	0068	1	:	V02-002	RPG002 Bob Grosso 7-Aug-1981
:	69	0069	1	:		lib\$gl_ctlmsk now a quadword
:	70	0070	1	:		
:	71	0071	1	:	V02-001	BLS0029 Benn Schreiber 23-Dec-1980
:	72	0072	1	:		Convert messages to message compiler. Libraries of shareable
:	73	0073	1	:		image symbol tables.
:	74	0074	1	:		
:	75	0075	1	:		
:	76	0076	1	:		
:	77	0077	1	:		

```

79      0078 1 LIBRARY
80      0079 1 'SYSS$LIBRARY:STARLET.L32';      !System data structures
81      0080 1 REQUIRE
82      0081 1 'PREFIX';                        !Librarian macros
83      0265 1 REQUIRE
84      0266 1 'LIBDEF';                        !Librarian data structures
85      0554 1 REQUIRE
86      0555 1 'LBRDEF';                        !Library processor definitions
87      1146 1
88      1147 1 EXTERNAL
89      1148 1     lbr$gl_rmsstv : ADDRESSING_MODE (GENERAL), !RMS STV from librarian
90      1149 1     lib$gl_ctlmsk : BLOCK [2],           !Library control mask
91      1150 1     lib$gl_libctl,                       !Library control index
92      1151 1     lib$gl_inpfdb : REF BBLOCK,          !Pointer to input FDB
93      1152 1     lib$gl_rab : BBLOCK,                 !Pointer to input RAB
94      1153 1     lib$gl_type,                          !Type of library opened
95      1154 1     lib$gl_inplist : REF BBLOCK;          !Listhead for input files
96      1155 1
97      1156 1 EXTERNAL ROUTINE
98      1157 1     find_list_width,                     !Determine listing width
99      1158 1     getfilnamdesc,                       !Get string descriptor for file name
100     1159 1     lib_get_mem,                          !Get memory
101     1160 1     lib_free_mem,                         !and give it back
102     1161 1     lib_get_zmem;                          !Get zeroed memory
103     1162 1
104     1163 1 EXTERNAL LITERAL
105     1164 1     lib$_initerr;                          !Error initializing library
```



```
107 1165 1 GLOBAL ROUTINE next_input_file =
108 1166 2 BEGIN
109 1167 2
110 1168 2 ++
111 1169 2
112 1170 2 FUNCTIONAL DESCRIPTION:
113 1171 2
114 1172 2 This routine opens the next input file and returns success or failure.
115 1173 2
116 1174 2 IMPLICIT INPUTS:
117 1175 2
118 1176 2 The list of input files is pointed to by lib$gl_inplist.
119 1177 2
120 1178 2 OUTPUT PARAMETERS:
121 1179 2 NONE
122 1180 2
123 1181 2 IMPLICIT OUTPUTS:
124 1182 2
125 1183 2 The file is opened, and lib$gl_inpfdb points to the current FDB.
126 1184 2
127 1185 2 --
128 1186 2
129 1187 2 LOCAL
130 1188 2 inpfab : BBLOCK [fab$cln]; !FAB to open file
131 1189 2
132 1190 2 IF .lib$gl_inpfdb EQL 0 !If fdb not set up yet
133 1191 2 THEN BEGIN
134 1192 2 lib$gl_inpfdb = lib$gl_inplist; !Then point at listhead to get first
135 1193 2 $RAB_INIT ( ! and initialize the RAB
136 1194 2 RAB = lib$al_rab,
137 1195 2 FAB = inpfab,
138 1196 2 ROP = LOC ! use locate mode
139 1197 2 );
140 1198 2
141 1199 2 ELSE BEGIN !Otherwise close open library
142 1200 2 LOCAL
143 1201 2 status;
144 1202 2 BIND
145 1203 2 inpdsc = lib$gl_inpfdb [fdb$l_namdesc] : BBLOCK;
146 1204 2
147 1205 2 $FAB_INIT (FAB = inpfab); !Initialize FAB as a FAB
148 1206 2 inpfab [fab$w_ifi] = .lib$gl_inpfdb [fdb$w_ifi]; !Set the IFI for CLOSE
149 1207 2 rms_perform ($DISCONNECT (RAB = lib$al_rab),
150 1208 2 lib$closein,
151 1209 2 .lib$al_rab [rab$l_stv], 1, inpdsc);
152 1210 2 rms_perform ($CLOSE (FAB = inpfab), !Close the library
153 1211 2 lib$closein,
154 1212 2 .inpfab [fab$l_stv], 1, inpdsc);
155 1213 2 END;
156 1214 2 lib$gl_inpfdb = .lib$gl_inpfdb [fdb$l_nxtfdb]; !Link to next FDB
157 1215 2 if .lib$gl_inpfdb EQL 0 !And if no more
158 1216 2 THEN BEGIN
159 1217 2 IF .lib$al_rab [rab$l_ubf] NEQ 0
160 1218 2 THEN lib_free_mem (lib$cl_maxrecsiz, .lib$al_rab [rab$l_ubf]);
161 1219 2 RETURN false
162 1220 2 END;
163 1221 2
```

```
164 1222 3 BEGIN
165 1223 3 LOCAL
166 1224 3 open_status;
167 1225 3
168 1226 3 BIND
169 1227 3 inpdesc = lib$gl_inpfdb [fdb$l_namdesc] : BBLOCK, !Descriptor for filename
170 1228 3 inpnam = lib$gl_inpfdb [fdb$t_nam] : BBLOCK; !Name NAM block
171 1229 3
172 P 1230 3 $FAB_INIT ( !Init the FAB
173 P 1231 3 FAB = inpfab,
174 P 1232 3 FAC = GET, !For gets
175 P 1233 3 MRS = lbr$c_maxrecsiz, !Max record size
176 P 1234 3 RFM = VAR, !Variable format records
177 P 1235 3 FOP = <NAM,SQO>, !Open by NAM block
178 P 1236 3 !Restrict to sequential access for network performance gain
179 P 1237 3 NAM = inpnam, !Point to the nam block
180 P 1238 3 FNS = .inpdesc [dsc$w_length], !Pass name also in case network open
181 P 1239 3 FNA = .inpdesc [dsc$a_pointer]
182 1240 3 );
183 1241 3 IF .lib$gl_type EQL lbr$c_ttyp_shstb !If a shareable image being opened
184 1242 3 THEN
185 1243 3 BEGIN
186 1244 3 inpfab [fab$v_bro] = true; ! then set bro also
187 1245 3 inpfab [fab$v_sqo] = false; ! and clear SQO
188 1246 3 END;
189 1247 3 open_status = $OPEN (FAB = inpfab);
190 1248 3 IF NOT .open_status
191 1249 3 THEN
192 1250 3 BEGIN
193 1251 3 SIGNAL (lib$openin,
194 1252 3 1, inpdesc,
195 1253 3 .inpfab [fab$l_stv],
196 1254 3 .open_status );
197 1255 3 RETURN .open_status;
198 1256 3 END;
199 1257 3 lib$gl_inpfdb [fdb$w_ifi] = .inpfab [fab$w_ifi]; !Save IFI for close
200 1258 3 IF .lib$al_rab [rab$_ubf] EQL 0 !If no user buffer allocated
201 1259 3 THEN BEGIN
202 P 1260 3 perform (lib$get_mem (lbr$c_maxrecsiz, lib$al_rab [rab$l_ubf]), !Allocate buffer
203 1261 3 lib$initerr, 1, inpdesc);
204 1262 3 lib$al_rab [rab$w_usz] = lbr$c_maxrecsiz; !and set size in RAB
205 1263 3 END;
206 P 1264 3 rms_perform ($CONNECT (RAB = lib$al_rab), !Connect for record I/O
207 P 1265 3 lib$openin,
208 1266 3 .lib$al_rab [rab$l_stv], 1, inpdesc);
209 1267 3 IF .lib$gl_type EQL lbr$c_ttyp_shstb !If opening a shareable image
210 1268 3 THEN BEGIN
211 1269 3 inpfab [fab$v_esc] = true; !Set the majik bit
212 1270 3 inpfab [fab$l_ctx] = rme$c_setrfm; !Function to set record format
213 1271 3 inpfab [fab$b_rfm] = fab$c_var; !Set variable length records for $gets
214 P 1272 3 rms_perform ($MODIFY (FAB = inpfab), !Tell RMS to think differently
215 P 1273 3 lib$openin, ! this should not fail
216 1274 3 inpfab [fab$l_stv], 1, inpdesc); ! but if it does report the error
217 1275 3 END;
218 1276 3 END;
219 1277 2 RETURN ss$_normal
220 1278 1 END; ! Of next_input_file
```



```
.TITLE  LIB FILEIO
.IDENT  \V04-000\
```

```
.EXTRN  LBR$GL_RMSSTV, LIB$GL_CTLMSK
.EXTRN  LIB$GL_LIBCTL, LIB$GL-INPFDB
.EXTRN  LIB$GL-RAB, LIB$GL TYPE
.EXTRN  LIB$GL-INPLIST, FIND LIST WIDTH
.EXTRN  GETFILNAMDESC, LIB GET MEM
.EXTRN  LIB FREE MEM, LIB GET ZMEM
.EXTRN  LIB$ INITERR, SYS$DISCONNECT
.EXTRN  SYS$CLOSE, SYS$OPEN
.EXTRN  SYS$CONNECT, SYS$MODIFY
```

```
.PSECT $CODE$,NOWRT,2
```

```
.ENTRY NEXT_INPUT_FILE, Save R2,R3,R4,R5,R6,R7,R8,-; 1165
R9-RT0;
```

```

MOVAB    LIB$GL_INPFDB, R10
MOVAB    LIB$SIGNAL, R9
MOVAB    $RMS_PTR, R8
MOVAB    -80(SP), SP
MOVL     LIB$GL_INPFDB, R6
BNEQ     1$
MOVAB    LIB$GL_INPLIST, LIB$GL_INPFDB
MOVCS    #0, (SP), #0, #68, $RMS_PTR

```

```
MOVW    #17409, $RMS_PTR      :  
MOVL    #65536, $RMS_PTR+4    :  
MOVAB   INPFAB, $RMS_PTR+60   :  
BRB     3$                    : 1190  
MOVAB   16(R6), R7            : 1203  
MOVCS   #0, (SP), #0, #80, $RMS_PTR : 1205
```

```
MOVW    #20483, $RMS_PTR
MOVW    #2, $RMS_PTR+22
MOVW    #2, $RMS_PTR+31
MOVW    6(R6), INPFAB+2
PUSHL   R8
```

```

PUSHL    R8
CALLS    #1, SYSSDISCONNECT
BLBS     STATUS, 2$
PUSHL    LIB$AL_RAB+12
PUSHL    STATUS
PUSHL    R7
PUSHL    #1
PUSHL    #8786000
CALLS    #5, LIB$SIGNAL
PUSHL    SP

```

```

PUSHL    #1, SYSSCLOSE
CALLS    STATUS, 3$
BLBS     INPFAB+12
PUSHL    STATUS
PUSHL    R7
PUSHL    #1
PUSHL    #8786000
CALLS    #5, LIB$SIGNAL

```

				07FC		00000	
			5A	0000G	CF	9E	00002
			59	00000000G	00	9E	00007
			58	0000G	CF	9E	0000E
			5E	B0	AE	9E	00013
			56		6A	DO	00017
					20	12	0001A
0044	8F	00	6A	0000G	CF	9E	0001C
			6E		00	2C	00021
					68		00028
			68	4401	8F	BO	00029
		04	A8	00010000	8F	DO	0002E
		3C	A8		6E	9E	00036
					5A	11	0003A
0050	8F	00	57	10	A6	9E	0003C
			6E		00	2C	00040
					6E		00047
			6E	5003	8F	BO	00048
		16	AE		02	90	0004D
		1F	AE		02	90	00051
		02	AE	06	A6	BO	00055
					58	DD	0005A
		00000000G	00		01	FB	0005C
			12		50	E8	00063
				0C	A8	DD	00066
					50	DD	00069
					57	DD	0006B
					01	DD	0006D
			69	00861050	8F	DD	0006F
					05	FB	00075
					5E	DD	00078
		00000000G	00		01	FB	0007A
			12		50	E8	00081
				0C	AE	DD	00084
					50	DD	00087
					57	DD	00089
					01	DD	0008B
			69	00861050	8F	DD	0008D
					05	FB	00093

0050	8F	56	7A	9A	DO	00096	3\$:	MOVL	@LIB\$GL_INPFDB, LIB\$GL_INPFDB	1214
		57	50	15	DO	00099		BNEQ	5\$	1215
		00		A8	DO	0009B		MOVL	LIB\$AL_RAB+36, R0	1217
				0C	13	0009F		BEQL	4\$	
				50	DD	000A1		PUSHL	R0	1218
			7E	8F	3C	000A3		MOVZWL	#2048, -(SP)	
		0000G	CF	02	FB	000A8		CALLS	#2, LIB_FREE_MEM	
				00F4	31	000AD	4\$:	BRW	12\$	1219
			6A	10	C1	000B0	5\$:	ADDL3	#16, LIB\$GL_INPFDB, R6	1227
			6A	8F	C1	000B4		ADDL3	#64, LIB\$GL_INPFDB, R7	1228
			6E	00	2C	000BC		MOVC5	#0, (SP), #0, #80, \$RMS_PTR	1240
				6E		000C3				
			6E	8F	B0	000C4		MOVW	#20483, \$RMS_PTR	
	04	AE	01000040	8F	DO	000C9		MOVL	#16777280, \$RMS_PTR+4	
	16	AE		02	90	000D1		MOVB	#2, \$RMS_PTR+22	
	28	AE		02	90	000D5		MOVB	#2, \$RMS_PTR+31	
	2C	AE		57	DO	000D9		MOVL	R7, \$RMS_PTR+40	
	34	AE	04	A6	DO	000DD		MOVL	4(R6), \$RMS_PTR+44	
	36	AE	0800	66	90	000E2		MOVB	(R6), \$RMS_PTR+52	
		05	0000G	8F	B0	000E6		MOVW	#2048, \$RMS_PTR+54	
				CF	D1	000EC		CMPL	LIB\$GL_TYPE, #5	1241
				0A	12	000F1		BNEQ	6\$	
	16	AE	40	8F	88	000F3		BISB2	#64, INPFAB+22	1244
	04	AE	40	8F	8A	000F8		BICB2	#64, INPFAB+4	1245
				5E	DD	000FD	6\$:	PUSHL	SP	1247
		00000000G	00	01	FB	000FF		CALLS	#1, SYSS\$OPEN	
			52	50	DO	00106		MOVL	R0, OPEN_STATUS	
			13	52	E8	00109		BLBS	OPEN_STATUS, 7\$	1248
				52	DD	0010C		PUSHL	OPEN_STATUS	1254
				56	DD	0010E		PUSHL	R6	1251
				01	DD	00110		PUSHL	#1	
			00861098	8F	DD	00112		PUSHL	#8786072	
	69			04	FB	00118		CALLS	#4, LIB\$SIGNAL	
	50			52	DO	0011B		MOVL	OPEN_STATUS, R0	1255
				04		0011E		RET		
	50			6A	DO	0011F	7\$:	MOVL	LIB\$GL_INPFDB, R0	1257
	06	A0	02	AE	B0	00122		MOVW	INPFAB+2, 6(R0)	
			24	A8	D5	00127		TSTL	LIB\$AL_RAB+36	1258
				25	12	0012A		BNEQ	9\$	
			24	A8	9F	0012C		PUSHAB	LIB\$AL_RAB+36	1261
			0800	8F	3C	0012F		MOVZWL	#2048, -(SP)	
		0000G	CF	02	FB	00134		CALLS	#2, LIB_GET_MEM	
			OF	50	E8	00139		BLBS	STATUS, 8\$	
				50	DD	0013C		PUSHL	STATUS	
				56	DD	0013E		PUSHL	R6	
				01	DD	00140		PUSHL	#1	
			00000000G	8F	DD	00142		PUSHL	#LIB\$ INITERR	
	69			04	FB	00148		CALLS	#4, LIB\$SIGNAL	
	20	A8	0800	8F	B0	0014B	8\$:	MOVW	#2048, LIB\$AL_RAB+32	1262
				58	DD	00151	9\$:	PUSHL	R8	1266
		00000000G	00	01	FB	00153		CALLS	#1, SYSS\$CONNECT	
			12	50	E8	0015A		BLBS	STATUS, 10\$	
				A8	DD	0015D		PUSHL	LIB\$AL_RAB+12	
				50	DD	00160		PUSHL	STATUS	
				56	DD	00162		PUSHL	R6	
				01	DD	00164		PUSHL	#1	
			00861098	8F	DD	00166		PUSHL	#8786072	

LIB FILE10
V04=000

H 1
16-Sep-1984 01:52:04
14-Sep-1984 12:38:03

VAX-11 Bliss-32 V4.0-742
[LIBRAR.SRC]FILE10.B32;1

Page 8
(3)

69	05	0000G	05	FB	0016C	CALLS	#5, LIB\$SIGNAL	1267	
05			CF	D1	0016F	10\$:	CMPL	LIB\$GL_TYPE, #5	1269
			2A	12	00174		BNEQ	11\$	1270
07	AE		08	88	00176		BISB2	#8, INPFAB+7	1271
18	AE		01	D0	0017A		MOVL	#1, INPFAB+24	1274
1F	AE		02	90	0017E		MOVB	#2, INPFAB+31	
			5E	DD	00182		PUSHL	SP	
00000000G	00		01	FB	00184		CALLS	#1, SYSS\$MODIFY	
	12		50	E8	0018B		BLBS	STATUS, 11\$	
		0C	AE	9F	0018E		PUSHAB	INPFAB+12	
			50	DD	00191		PUSHL	STATUS	
			56	DD	00193		PUSHL	R6	
			01	DD	00195		PUSHL	#1	
		00861098	8F	DD	00197		PUSHL	#8786072	
69	05		05	FB	0019D		CALLS	#5, LIB\$SIGNAL	1277
50	01		01	D0	001A0	11\$:	MOVL	#1, R0	1278
				04	001A3		RET		
			50	D4	001A4	12\$:	CLRL	R0	
				04	001A6		RET		

; Routine Size: 423 bytes, Routine Base: \$CODE\$ + 0000

```
1279 1 GLOBAL ROUTINE get_record (record_desc) =
1280 2 BEGIN
1281 2
1282 2 FUNCTIONAL DESCRIPTION:
1283 2
1284 2 This routine reads the next record from the current input file
1285 2
1286 2 Inputs:
1287 2
1288 2 NONE
1289 2
1290 2 Implicit inputs:
1291 2
1292 2 Input file must be open
1293 2
1294 2 Outputs:
1295 2
1296 2 record_desc is a string descriptor for the record
1297 2
1298 2 --
1299 2 LOCAL
1300 2 status;
1301 2
1302 2 MAP
1303 2 record_desc : REF BBLOCK;
1304 2
1305 2 BIND
1306 2 inpdesc = lib$gl_inpfdb [fdb$l_namdesc] : BBLOCK;
1307 2
1308 2 status = $GET (RAB = lib$al_rab); !Read the record
1309 2 IF NOT .status AND (.status NEQ rms$eof)
1310 2 THEN SIGNAL (lib$readerr OR sts$severe,
1311 2 1, inpdesc, .status, .lib$al_rab [rab$l_stv]);
1312 2 record_desc [dsc$w_length] = .lib$al_rab [rab$w_rs2]; !Return length
1313 2 record_desc [dsc$a_pointer] = .lib$al_rab [rab$w_rbf]; !and address of record
1314 2 RETURN .status
1315 1 END; ! Of get_record
```

52	0000G	CF		0000G	10	C1	00002		.EXTRN	SYSSGET	
					CF	9F	00008		.ENTRY	GET_RECORD, Save R2,R3	1279
	00000000G	00			01	FB	0000C		ADDL3	#16, LIB\$GL_INPFDB, R2	1306
		53			50	D0	00013		PUSHAB	LIB\$AL_RAB	1308
		1E			53	E8	00016		CALLS	#1, SYSSGET	
	0001827A	8F			53	D1	00019		MOVL	R0, STATUS	1309
					15	13	00020		BLBS	STATUS, 1\$	
				0000G	CF	DD	00022		CMPL	STATUS, #98938	
					0C	BB	00026		BEQL	1\$	
					01	DD	00028		PUSHL	LIB\$AL_RAB+12	1311
					8F	DD	0002A		PUSHR	#*M<R2,R3>	1310
	00000000G	00		008610B6	05	FB	00030		PUSHL	#1	
		50			05	FB	00030		PUSHL	#8786102	
		60			04	AC	00037	1\$:	CALLS	#5, LIB\$SIGNAL	
				0000G	CF	B0	0003B		MOVL	RECORD_DESC, R0	1312
									MOVW	LIB\$AL_RAB+34, (R0)	

LIB_FILE10
V04=000

J 1
16-Sep-1984 01:52:04 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:38:03 [LIBRAR.SRC]FILE10.B32;1

Page 10
(4)

04 A0 0000G CF D0 00040
50 DO 00046
04 00049

MOVL LIB\$AL_RAB+40, 4(R0)
MOVL STATUS, R0
RET

: 1313
: 1314
: 1315

; Routine Size: 74 bytes, Routine Base: \$CODE\$ + 01A7

```
260 1316 1 GLOBAL ROUTINE lib_open_out (fdb, relnam, carriage_cntrl, listingwidth) =
261 1317 BEGIN
262 1318
263 1319 ++
264 1320 FUNCTIONAL DESCRIPTION:
265 1321 This routine opens the output file specified by fdb.
266 1322
267 1323 INPUTS:
268 1324
269 1325 fdb address of the file descriptor block
270 1326 relnam Address of related NAM block
271 1327 carriage_cntrl true if fab$u_cr, false if not
272 1328 listingwidth optional address of place to store width of device
273 1329
274 1330 OUTPUTS:
275 1331
276 1332 fdb[fdb$w_ifi] iff of open file
277 1333
278 1334 --
279 1335
280 1336 MAP
281 1337 fdb : REF BBLOCK;
282 1338
283 1339 LOCAL
284 1340 status,
285 1341 ofab : BBLOCK [fab$c_bln];
286 1342
287 1343 BIND
288 1344 namblk = fdb [fdb$t_nam] : BBLOCK, !Name the NAM block
289 1345 filedefdesc = fdb [fdb$t_defext] : BBLOCK, !String descriptor for default filename
290 1346 filenamedesc = fdb [fdb$t_namdesc] : BBLOCK; !String descriptor for filename
291 1347
292 1348 BUILTIN
293 1349 NULLPARAMETER;
294 1350
295 1351 $FAB_INIT ( !Initialize the FAB
296 1352 FAB = ofab,
297 1353 RFM = VAR,
298 1354 FNS = .filenameesc [dsc$w_length],
299 1355 FNA = .filenameesc [dsc$a_pointer],
300 1356 DNS = .filedefdesc [dsc$w_length],
301 1357 DNA = .filedefdesc [dsc$a_pointer],
302 1358 FAC = PUT,
303 1359 NAM = namblk,
304 1360 FOP = <OFP, $QO> !Restrict to sequential access for network performance gain
305 1361 );
306 1362
307 1363 IF .carriage_cntrl !If carriage control desired
308 1364 THEN ofab [fab$u_cr] = true;
309 1365
310 1366 $RAB_INIT ( !Initialize the RAB
311 1367 RAB = lib$a1_rab,
312 1368 FAB = ofab
313 1369 );
314 1370 namblk [nam$l_rlf] = .relnam; !Set address of related NAM block
315 1371
316 1372 ! Create the file
```

```
317 1373 2 !
318 1374 2 status = $CREATE (FAB = ofab); !Create the file
319 1375 2 IF NOT .status !If error
320 1376 2 THEN BEGIN
321 1377 2     getfilnamdesc (ofab, filenamedesc); !Get string descriptor for name
322 1378 2     SIGNAL_STOP ( !Signal error and stop
323 1379 2         lib$openout,
324 1380 2         1, filenamedesc, .status, .ofab [fab$l_stv]);
325 1381 2     END;
326 1382 2 getfilnamdesc (ofab, filenamedesc);
327 1383 2 rms_perform ($CONNECT (RAB = lib$al_rab), !Connect the record stream
328 1384 2     lib$openout,
329 1385 2     .lib$al_rab [rab$l_stv], 1, filenamedesc);
330 1386 2
331 1387 2 Set it into fdb and return
332 1388 2
333 1389 2 fdb [fdb$w_ifi] = .ofab [fab$w_ifi];
334 1390 2 IF NOT NUL[PARAMETER (4)
335 1391 2 THEN perform (find_list_width (ofab, .listingwidth));
336 1392 2 RETURN true
337 1393 1 END; !Of lib_open_out
```

.EXTRN SYSS\$CREATE

.ENTRY LIB_OPEN_OUT, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 1316

R10

MOVAB \$RMS_PTR, R10

MOVAB -80(SP), SP

MOVL FDB, R6 1344

MOVAB 64(R6), R9

MOVAB 8(R6), R7 1345

MOVAB 16(R6), R8 1346

MOVCS #0, (SP), #0, #80, \$RMS_PTR 1361

MOVW #20483, \$RMS_PTR

MOVL #536870976, \$RMS_PTR+4

MOVB #1, \$RMS_PTR+22

MOVB #2, \$RMS_PTR+31

MOVL R9, \$RMS_PTR+40

MOVL 4(R8), \$RMS_PTR+44

MOVL 4(R7), \$RMS_PTR+48

MOVB (R8), \$RMS_PTR+52

MOVB (R7), \$RMS_PTR+53

BLBC CARRIAGE_CNTRL, 1\$ 1363

BISB2 #2, OFAB+30 1364

MOVCS #0, (SP), #0, #68, \$RMS_PTR 1369

MOVW #17409, \$RMS_PTR

MOVAB OFAB, \$RMS_PTR+60

MOVL RELNAM, 16(R9) 1370

PUSHL SP 1374

CALLS #1, SYSS\$CREATE

MOVL R0, STATUS

BLBS STATUS, 2\$ 1375

PUSHL R8 1377

				07FC 00000
	SA	0000G	CF	9E 00002
	5E	B0	AE	9E 00007
	56	04	AC	D0 0000B
	59	40	A6	9E 0000F
	57	08	A6	9E 00013
	58	10	A6	9E 00017
0050	8F	00	00	2C 0001B
	6E		6E	00022
	6E	5003	8F	B0 00023
	04	AE 20000040	8F	D0 00028
	16	AE	01	90 00030
	1F	AE	02	90 00034
	28	AE	59	D0 00038
	2C	AE	A8	D0 0003C
	30	AE	A7	D0 00041
	34	AE	68	90 00046
	35	AE	67	90 0004A
	04	0C	AC	E9 0004E
0044	8F	00	02	88 00052
	6E		00	2C 00056
	6A	4401	6A	0005D
	3C	AA	8F	B0 0005E
	10	A9	6E	9E 00063
		08	AC	D0 00067
	00000000G	00	5E	DD 0006C
		52	01	FB 0006E
		20	50	D0 00075
			52	E8 00078
			58	DD 0007B

0000G	CF	04	AE	9F	0007D	PUSHAB	OFAB		
		0C	02	FB	00080	CALLS	#2, GETFILNAMDESC		
			AE	DD	00085	PUSHL	OFAB+12	1380	
			52	DD	00088	PUSHL	STATUS		
			58	DD	0008A	PUSHL	R8	1378	
			01	DD	0008C	PUSHL	#1		
		008610A4	8F	DD	0008E	PUSHL	#8786084		
00000000G	00		05	FB	00094	CALLS	#5, LIB\$STOP		
			58	DD	0009B	PUSHL	R8	1382	
		04	AE	9F	0009D	PUSHAB	OFAB		
0000G	CF		02	FB	000A0	CALLS	#2, GETFILNAMDESC		
			5A	DD	000A5	PUSHL	R10	1385	
00000000G	00		01	FB	000A7	CALLS	#1, SYS\$CONNECT		
	16		50	EB	000AE	BLBS	STATUS, 3\$		
		0C	AA	DD	000B1	PUSHL	LIB\$AL_RAB+12		
			50	DD	000B4	PUSHL	STATUS		
			58	DD	000B6	PUSHL	R8		
			01	DD	000B8	PUSHL	#1		
		008610A4	8F	DD	000BA	PUSHL	#8786084		
00000000G	00		05	FB	000C0	CALLS	#5, LIB\$SIGNAL		
	06		AE	B0	000C7	MOVW	OFAB+2, 6(R6)	1389	
		02	6C	91	000CC	CMPB	(AP), #4	1390	
			13	1F	000CF	BLSSU	4\$		
		10	AC	D5	000D1	TSTL	16(AP)		
			0E	13	000D4	BEQL	4\$		
		10	AC	DD	000D6	PUSHL	LISTINGWIDTH	1391	
		04	AE	9F	000D9	PUSHAB	OFAB		
0000G	CF		02	FB	000DC	CALLS	#2, FIND_LIST_WIDTH		
	03		50	E9	000E1	BLBC	STATUS, 5\$		
	50		01	D0	000E4	MOVL	#1, R0	1392	
			04	000E7	5\$: RET			1393	

; Routine Size: 232 bytes, Routine Base: \$CODE\$ + 01F1

```
339 1394 1 GLOBAL ROUTINE lib_close_out (fdb, delete) =
340 1395 2 BEGIN
341 1396 2 ++
342 1397 2 Close the open output file
343 1398 2
344 1399 2 Inputs:
345 1400 2
346 1401 2 fdb      Address of the fdb for the file
347 1402 2 delete  True to delete file
348 1403 2
349 1404 2 Outputs:
350 1405 2
351 1406 2 file is closed
352 1407 2
353 1408 2 --
354 1409 2
355 1410 2 MAP
356 1411 2 fdb : REF BBLOCK;
357 1412 2
358 1413 2 BIND
359 1414 2 namblk = fdb [fdb$st_nam] : BBLOCK;
360 1415 2
361 1416 2 LOCAL
362 1417 2 ofab : BBLOCK [fab$c_bln];
363 1418 2
364 1419 2 $FAB_INIT (FAB = ofab); !Make a FAB
365 1420 2 ofab [fab$u_dlt] = .delete; !Set delete flag true/false
366 1421 2 ofab [fab$u_ifi] = .fdb [fdb$u_ifi]; !Set IFI for close
367 1422 2 ofab [fab$l_nam] = namblk;
368 1423 2 lib$al_rab [rab$l_fab] = ofab;
369 1424 2 rms_perform ($DISCONNECT (RAB = lib$al_rab), !Disconnect record stream
370 1425 2 lib$closeout,
371 1426 2 .lib$al_rab [rab$l_stv], 1, fdb [fdb$l_namdesc]);
372 1427 2 rms_perform ($CLOSE (FAB = ofab), !Close the file
373 1428 2 lib$closeout,
374 1429 2 .ofab [fab$l_stv], 1, fdb [fdb$l_namdesc]);
375 1430 2 RETURN true
376 1431 1 END;
```

0050	8F	00	57	00000000G	00	9E	00002	.ENTRY	LIB CLOSE OUT, Save R2,R3,R4,R5,R6,R7	1394
			5E	B0	AE	9E	00009	MOVAB	LIB\$SIGNAC, R7	
			56	04	AC	D0	0000D	MOVAB	-80(SP), SP	1414
			6E		00	2C	00011	MOVL	FDB, R6	1419
					6E		00018	MOVC5	#0, (SP), #0, #80, \$RMS_PTR	
			6E	5003	8F	B0	00019	MOVW	#20483, \$RMS_PTR	
	16		AE		02	90	0001E	MOVB	#2, \$RMS_PTR+22	
	1F		AE		02	90	00022	MOVB	#2, \$RMS_PTR+31	
05	AE	01	07	08	AC	F0	00026	INSV	DELETE, #7, #1, OFAB+5	1420
			02	06	A6	B0	0002D	MOVW	6(R6), OFAB+2	1421
			28	40	A6	9E	00032	MOVAB	64(R6), OFAB+40	1422
			0000G	CF	6E	9E	00037	MOVAB	OFAB, LIB\$AL_RAB+60	1423
				0000G	CF	9F	0003C	PUSHAB	LIB\$AL_RAB	1426

```
00000000G 00 01 FB 00040 CALLS #1, SYSSDISCONNECT
14 50 EB 00047 BLBS STATUS, 1$
0000G CF DD 0004A PUSHL LIB$AL_RAB+12
50 DD 0004E PUSHL STATUS
10 A6 9F 00050 PUSHAB 16(R6)
01 DD 00053 PUSHL #1
67 00861058 8F DD 00055 PUSHL #8786008
05 FB 00058 CALLS #5, LIB$SIGNAL
5E DD 0005E 1$: PUSHL SP
01 FB 00060 CALLS #1, SYSSCLOSE
50 EB 00067 BLBS STATUS, 2$
0C AE DD 0006A PUSHL OFAB+12
50 DD 0006D PUSHL STATUS
10 A6 9F 0006F PUSHAB 16(R6)
01 DD 00072 PUSHL #1
67 00861058 8F DD 00074 PUSHL #8786008
05 FB 0007A CALLS #5, LIB$SIGNAL
50 01 DD 0007D 2$: MOVL #1, R0
04 00080 RET
```

1429

1430
1431

; Routine Size: 129 bytes, Routine Base: \$CODE\$ + 02D9

```
: 377 1432 1 END ! Of module
: 378 1433 0 ELUDOM
```

.EXTRN LIB\$SIGNAL, LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	858	NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)

Library Statistics

File	Symbols		Pages Mapped	Processing Time
	Total	Loaded Percent		
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	110 1	581	00:01.0

COMMAND QUALIFIERS

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS:FILE10/OBJ=OBJ\$:FILE10 MSRC\$:FILE10/UPDATE=(ENH\$:FILE10)

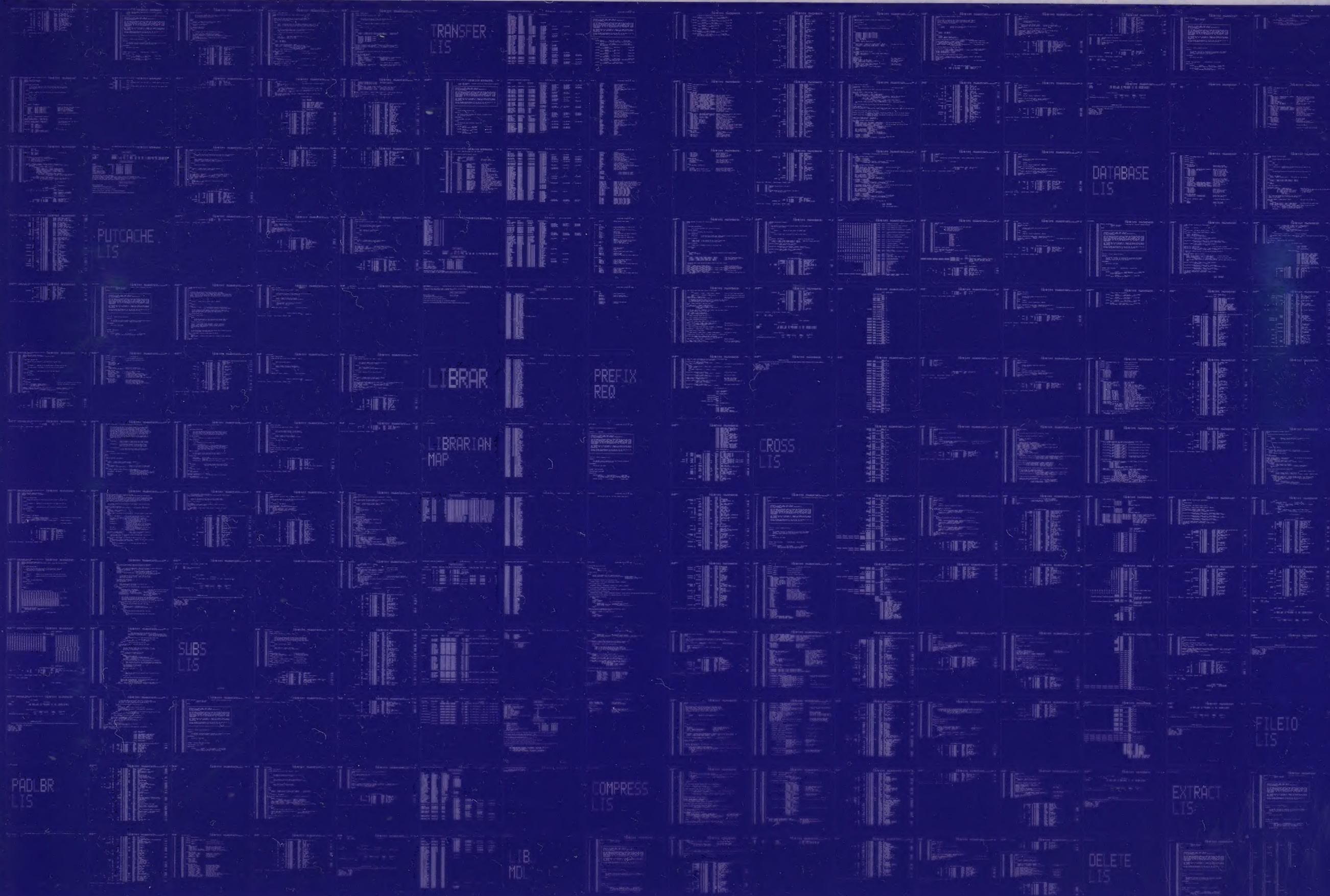
LIB FILEIO
V04=000

C²
16-Sep-1984 01:52:04

VAX-11 BLISS-32 V4.0-742

Page 16

: Size: 858 code + 0 data bytes
: Run Time: 00:30.3
: Elapsed Time: 00:59.8
: Lines/CPU Min: 2840
: Lexemes/CPU-Min: 52015
: Memory Used: 252 pages
: Compilation Complete



0201 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY